iOS Programmierung

(mit Swift)

Peter Braun, Florian Bachmann & Andreas Wittmann
@pe_braun @florianbachmann @anwittmann

Deutsche Telekom AG
FHWS - Hochschule für angewandte Wissenschaften Würzburg-Schweinfurt

#FHWSSwift

Agenda

1. **Introduction** – Organisatorisches
2. **First iOS-Project** – Hello World, **First iOS-Project** – Still Hello World (now with Code 😄)
3. **Swift**, Wait!, What about Objective-C?, Why Swift?
4. **A (not so) Quick Tour**
5. **Documentation**
6. **The basics –** iOS Architecture & more
7. **User Interfaces –** View Controller, Auto Layout & Size Classes
8. **Storyboard & Segues**
9. **Tables & NavigationController**
10. **TabBarController**
11. **Notifications**
12. **PickerViews**
13. **Touches, Gestures, 3D Touch, Peek & Pop**
14. **ScrollView & StackViews**
15. **Networking** – JSON & Dependency Managers
16. **WebKit**
17. **Maps**
18. **Storage & Data persistency –** NSUserDefaults, NSKeyedArchiver & Core Data
19. **ObjC**

Today

- Networking
  - Why NSURLSession? And not the old stuff
- Working with JSON
  - Plain Swift
  - SwiftyJSON
  - Example
    - OpenWeather API
- 3rd Party Code - Using Dependency Managers
  - ‘Copy & Paste’
  - CocoaPods,
  - Carthage

Networking

Why NSURLSession? And not the old stuff

NSURLSession

- NSURLSession is the ‘successor’ of NSURLConnection
- NSURLSession offers:
  - Background uploads and downloads
  - Pause and Resume Downloading
  - Configurable container
  - Rich delegate model

NSURLSession

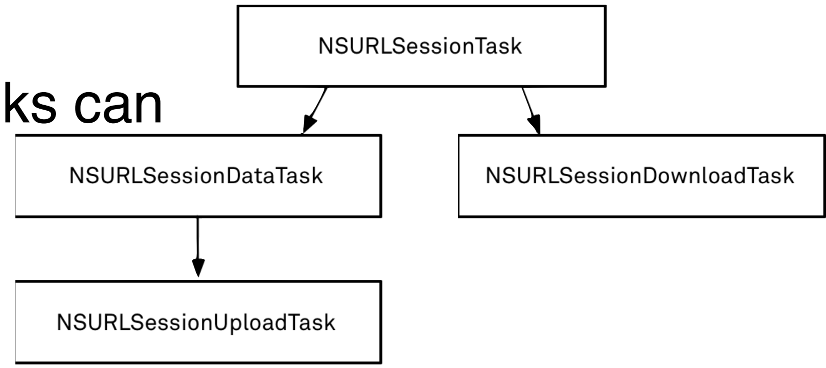
NSURLSession is the ‘successor’ of NSURLConnection

- There are 3 types of NSURLSession:

1. **default sessions**: behavior like NSURLConnection
2. **ephemeral sessions**: not cache any content to disk
3. **download sessions**: store the result in file and transferring data even when app is suspended, exits or crashes

- Based on above sessions, 3 types types of tasks can be scheduled:

1. **data tasks**: retrieve data to memory
2. **download tasks**: download file to disk
3. **upload tasks**: uploading file from disk and receiving response as data in memory



objc.io: Issue#5

NSURLConnection (old)

```
 let wuerzburgWeatherURL:NSURL  = ...

//generate NSURLRequest from URL
var requestWeather: NSURLRequest = NSURLRequest(URL: wuerzburgWeatherURL)
//generate queue for datahandling from Callback
let queue:NSOperationQueue = NSOperationQueue()

//Always sendAsynchronous don’t block the UI!
NSURLConnection.sendAsynchronousRequest(requestWeather, queue: queue, completionHandler:{
  (response: NSURLResponse!, data: NSData!, error: NSError!) -> Void in
  //Error handling!
  var err: NSError
  //Parse Result to json
  var jr: NSDictionary = NSJSONSerialization.
                         JSONObjectWithData(data, options:
                         NSJSONReadingOptions.MutableContainers, error: nil) as NSDictionary
  print(jr)
})
```

NSURLSession

```
 let wuerzburgWeatherURL:NSURL  = ...

var request: NSURLRequest = NSURLRequest(URL:wuerzburgWeatherURL)

//set HTTP Header Configurations
let config = NSURLSessionConfiguration.defaultSessionConfiguration()
let session = NSURLSession(configuration: config)

session.dataTaskWithRequest(request, completionHandler: {
    (data, response, error) in

    //Parse Result to json
    var jr: NSDictionary = NSJSONSerialization.
                           JSONObjectWithData(data, options:
                           NSJSONReadingOptions.MutableContainers, error: nil) as NSDictionary
    print(jr)

}).resume()
```

Working with JSON

Plain Swift
& SwiftyJSON

Parse JSON to Swift Types 1/6

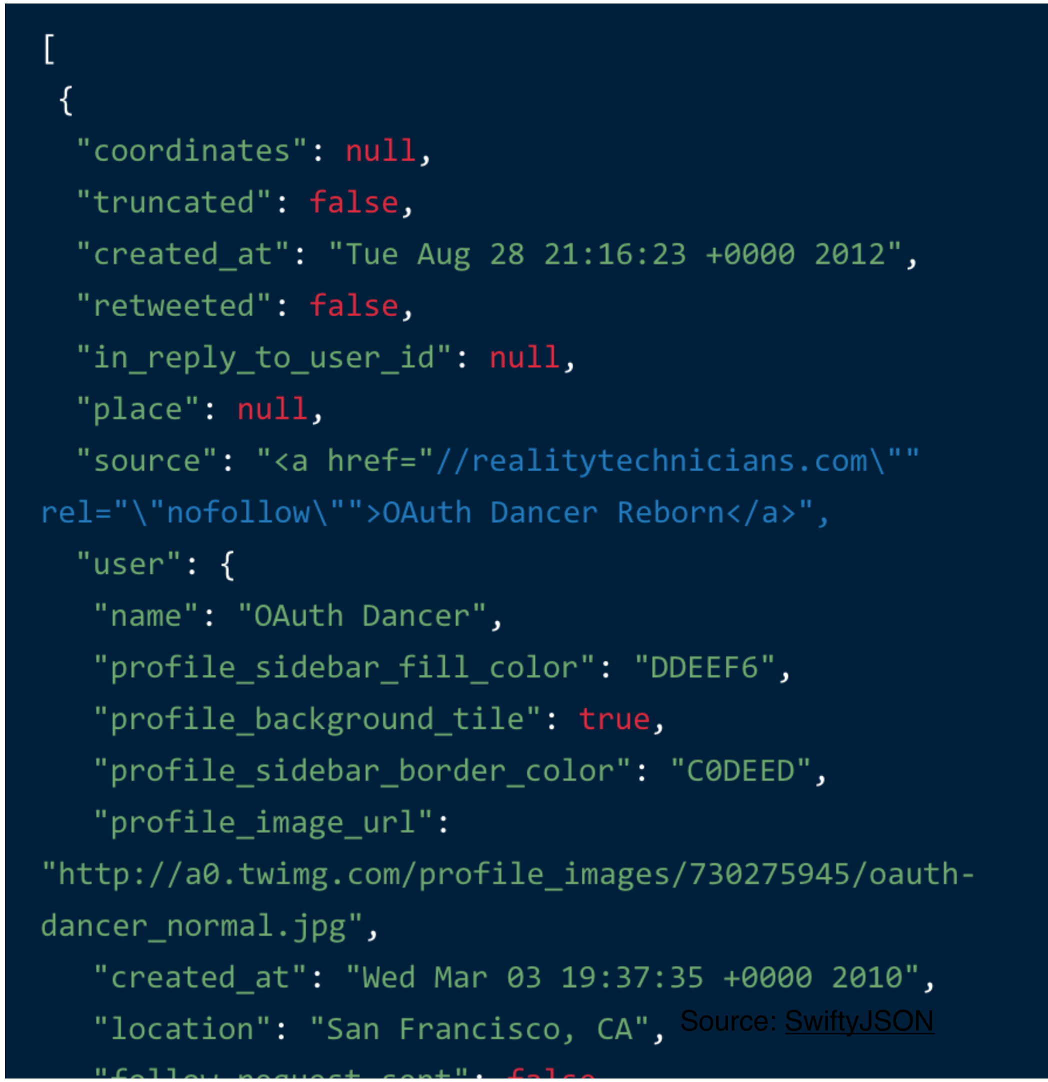
- Swift is very strict about types.
- In most cases, that is desired behaviour
- but it is a pain in the ass dealing with JSON data (remember the weather example)

Source: SwiftyJSON

Parse JSON to Swift Types 2/6

- **Example**:
  This Twitter API
  https://dev.twitter.com/docs/api/1.1/get/statuses/home_timeline
  returns a collections of the most recent tweets

- **Mission**:
  We want to get the users corresponding to each tweet



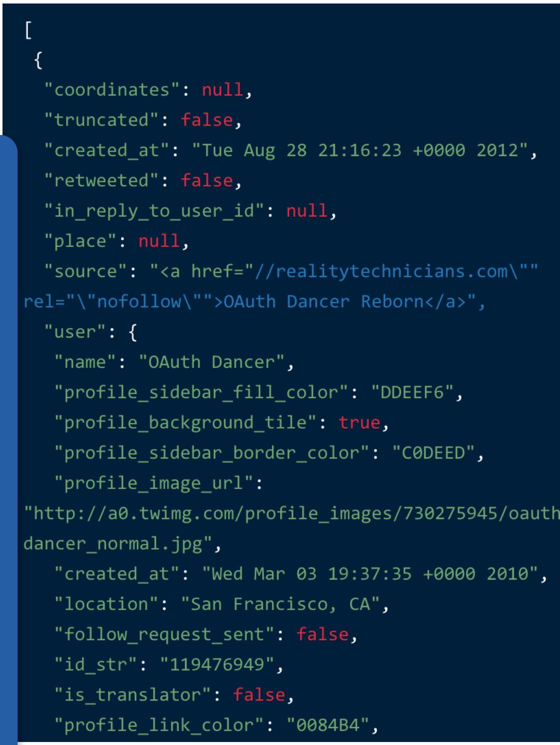
```
[
 {
  "coordinates": null,
  "truncated": false,
  "created_at": "Tue Aug 28 21:16:23 +0000 2012",
  "retweeted": false,
  "in_reply_to_user_id": null,
  "place": null,
  "source": "<a href="//realitytechnicians.com\""
rel="\"nofollow\"">OAuth Dancer Reborn</a>",
  "user": {
   "name": "OAuth Dancer",
   "profile_sidebar_fill_color": "DDEEF6",
   "profile_background_tile": true,
   "profile_sidebar_border_color": "C0DEED",
   "profile_image_url":
"http://a0.twimg.com/profile_images/730275945/oauth-
dancer_normal.jpg",
   "created_at": "Wed Mar 03 19:37:35 +0000 2010",
   "location": "San Francisco, CA",
   "follow_request_sent": false
```

Source: SwiftyJSON

Parse JSON to Swift Types 3/6

The code would look like this:

```
 let jsonObject : AnyObject! =
NSJSONSerialization.JSONObjectWithData(dataFromTwitter, options:
NSJSONReadingOptions.MutableContainers, error: nil)


if let statusesArray = jsonObject as? NSArray{
    if let aStatus = statusesArray[0] as? NSDictionary{
        if let user = aStatus["user"] as? NSDictionary{
            if let userName = user["name"] as? NSDictionary{
                print("yeah the name: \(userName)")
            }
        }
    }
}
```



```
[
 {
  "coordinates": null,
  "truncated": false,
  "created_at": "Tue Aug 28 21:16:23 +0000 2012",
  "retweeted": false,
  "in_reply_to_user_id": null,
  "place": null,
  "source": "<a href="//realitytechnicians.com\""
rel="\"nofollow\"">OAuth Dancer Reborn</a>",
  "user": {
   "name": "OAuth Dancer",
   "profile_sidebar_fill_color": "DDEEF6",
   "profile_background_tile": true,
   "profile_sidebar_border_color": "C0DEED",
   "profile_image_url":
"http://a0.twimg.com/profile_images/730275945/oauth
dancer_normal.jpg",
   "created_at": "Wed Mar 03 19:37:35 +0000 2010",
   "location": "San Francisco, CA",
   "follow_request_sent": false,
   "id_str": "119476949",
   "is_translator": false,
   "profile_link_color": "0084B4",
```

Parse JSON to Swift Types 4/6

Optional chaining would make it a bit better

```
 let jsonObject : AnyObject! =
NSJSONSerialization.JSONObjectWithData(dataFromTwitter, options:
NSJSONReadingOptions.MutableContainers, error: nil)


if let userName = (((jsonObject as? NSArray)?[0] as?
NSDictionary)?["user"] as? NSDictionary)?["name"]{
   //What A disaster above
   print("yeah the name: \(userName)”)
}
```

```
[
 {
  "coordinates": null,
  "truncated": false,
  "created_at": "Tue Aug 28 21:16:23 +0000 2012",
  "retweeted": false,
  "in_reply_to_user_id": null,
  "place": null,
  "source": "<a href="//realitytechnicians.com\""
rel="\"nofollow\"">OAuth Dancer Reborn</a>",
  "user": {
   "name": "OAuth Dancer",
   "profile_sidebar_fill_color": "DDEEF6",
   "profile_background_tile": true,
   "profile_sidebar_border_color": "C0DEED",
   "profile_image_url":
"http://a0.twimg.com/profile_images/730275945/oauth
dancer_normal.jpg",
   "created_at": "Wed Mar 03 19:37:35 +0000 2010",
   "location": "San Francisco, CA",
   "follow_request_sent": false,
   "id_str": "119476949",
   "is_translator": false,
   "profile_link_color": "0084B4",
```

Parse JSON to Swift Types 4/6

parsing JSON 2.0 style with Guard

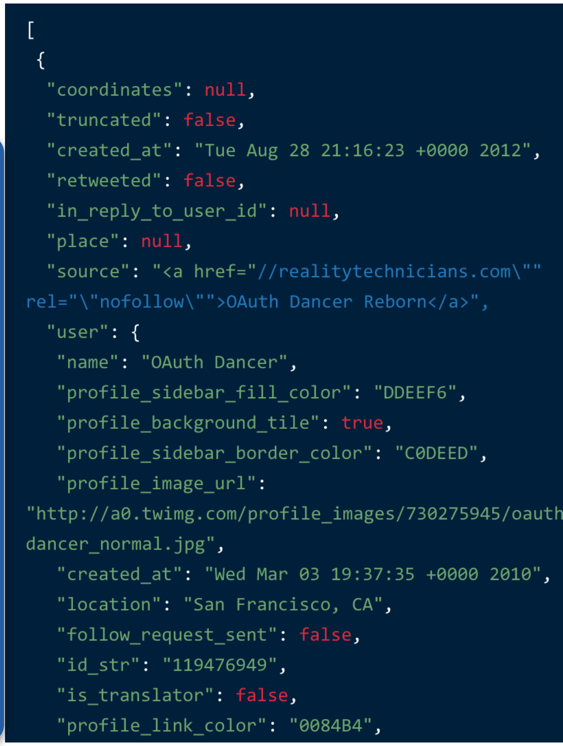
```
 let jsonObject : AnyObject! =
NSJSONSerialization.JSONObjectWithData(dataFromTwitter, options:
NSJSONReadingOptions.MutableContainers, error: nil)


guard let statusesArray = jsonObject as? NSArray else {
    print("failed to parse 'JSONObject'")
    return
}
guard let aStatus = statusesArray[0] as? NSDictionary else {
    print("failed to parse 'aStatus'")
    return
}
guard let user = aStatus[user] as? NSDictionary else {
    print("failed to parse 'aStatus'")
    return
}
guard if let userName = user["name"] as? NSDictionary{
             print("failed to parse 'userName'")
    return
 }
  print(userName)
}
```

```
[
 {
  "coordinates": null,
  "truncated": false,
  "created_at": "Tue Aug 28 21:16:23 +0000 2012",
  "retweeted": false,
  "in_reply_to_user_id": null,
  "place": null,
  "source": "<a href="//realitytechnicians.com\""
rel="\"nofollow\"">OAuth Dancer Reborn</a>",
  "user": {
   "name": "OAuth Dancer",
   "profile_sidebar_fill_color": "DDEEF6",
   "profile_background_tile": true,
   "profile_sidebar_border_color": "C0DEED",
   "profile_image_url":
"http://a0.twimg.com/profile_images/730275945/oauth
dancer_normal.jpg",
   "created_at": "Wed Mar 03 19:37:35 +0000 2010",
   "location": "San Francisco, CA",
   "follow_request_sent": false,
   "id_str": "119476949",
   "is_translator": false,
   "profile_link_color": "0084B4",
```

SwiftyJSON* 5/6

with SwiftyJSON:

```
import SwiftyJSON

...

let json = JSON(data: dataFromTwitter)
if let userName = json[0]["user"]["name"].string{
  //Now you got your value
  print("yeah the name: \(userName)”)
}
```



```
[
 {
  "coordinates": null,
  "truncated": false,
  "created_at": "Tue Aug 28 21:16:23 +0000 2012",
  "retweeted": false,
  "in_reply_to_user_id": null,
  "place": null,
  "source": "<a href="//realitytechnicians.com\""
rel="\"nofollow\"">OAuth Dancer Reborn</a>",
  "user": {
   "name": "OAuth Dancer",
   "profile_sidebar_fill_color": "DDEEF6",
   "profile_background_tile": true,
   "profile_sidebar_border_color": "C0DEED",
   "profile_image_url":
"http://a0.twimg.com/profile_images/730275945/oauth
dancer_normal.jpg",
   "created_at": "Wed Mar 03 19:37:35 +0000 2010",
   "location": "San Francisco, CA",
   "follow_request_sent": false,
   "id_str": "119476949",
   "is_translator": false,
   "profile_link_color": "0084B4",
```

* needs to be imported into the project via ‘Copy/Paste’, CocoaPods or Carthage. Details in a couple of slides

Example

OpenWeather API

Example

```
//current conditions
{"id":88319,"dt":1345284000,"name":"Benghazi",
    "coord":{"lat":32.12,"lon":20.07},
    "main":{"temp":306.15,"pressure":1013,"humidity":44,"temp_min":306,"temp_max":306},
    "wind":{"speed":1,"deg":-7},
    "weather":[
                 {"id":520,"main":"rain","description":"light intensity shower
rain","icon":"09d"},
                 {"id":500,"main":"rain","description":"light rain","icon":"10d"},
                 {"id":701,"main":"mist","description":"mist","icon":"50d"}
              ],
    "clouds":{"all":90},
    "rain":{"3h":3}}
```

Example

API Documentation:
http://openweathermap.org/weather-data#current

3rd Party Code

And Dependency Managers like
CocoaPods,
Carthage
& ‘Copy & Paste’

Dependency Managers

- Copy & Paste
  - Easy!
  - but doesn't update itself
- Dependency Managers are easy to try or integrate 3rd party code

⟨COCOAPODS⟩	http://cocoapods.org/
Carthage	https://github.com/Carthage/Carthage

⟨COCOAPODS⟩ Carthage

- Both are Cocoa (iOS/ OS X) dependency mangers
- CocoaPods
  - is a long-standing dependency manager for Cocoa.
  - has a centralized Repository
  - “Ultimately, the goal is to improve discoverability of, and engagement in, third party open-source libraries, by creating a more centralized ecosystem.”
- Carthage
  - “Carthage is intended to be the simplest way to add frameworks to your Cocoa application.”

CocoaPods

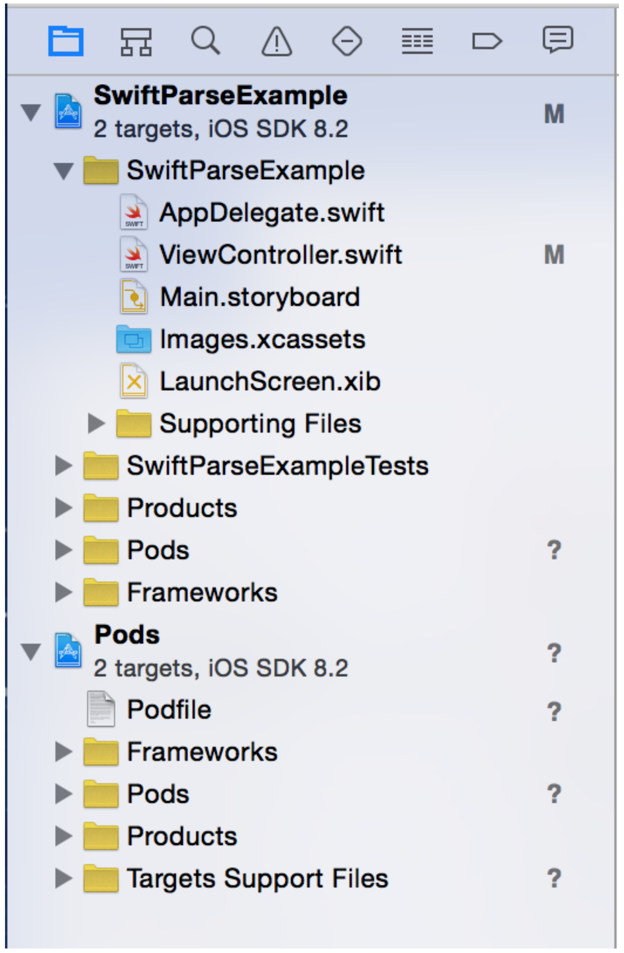
- A Podfile file describes the used libraries

- Example Cartfile:

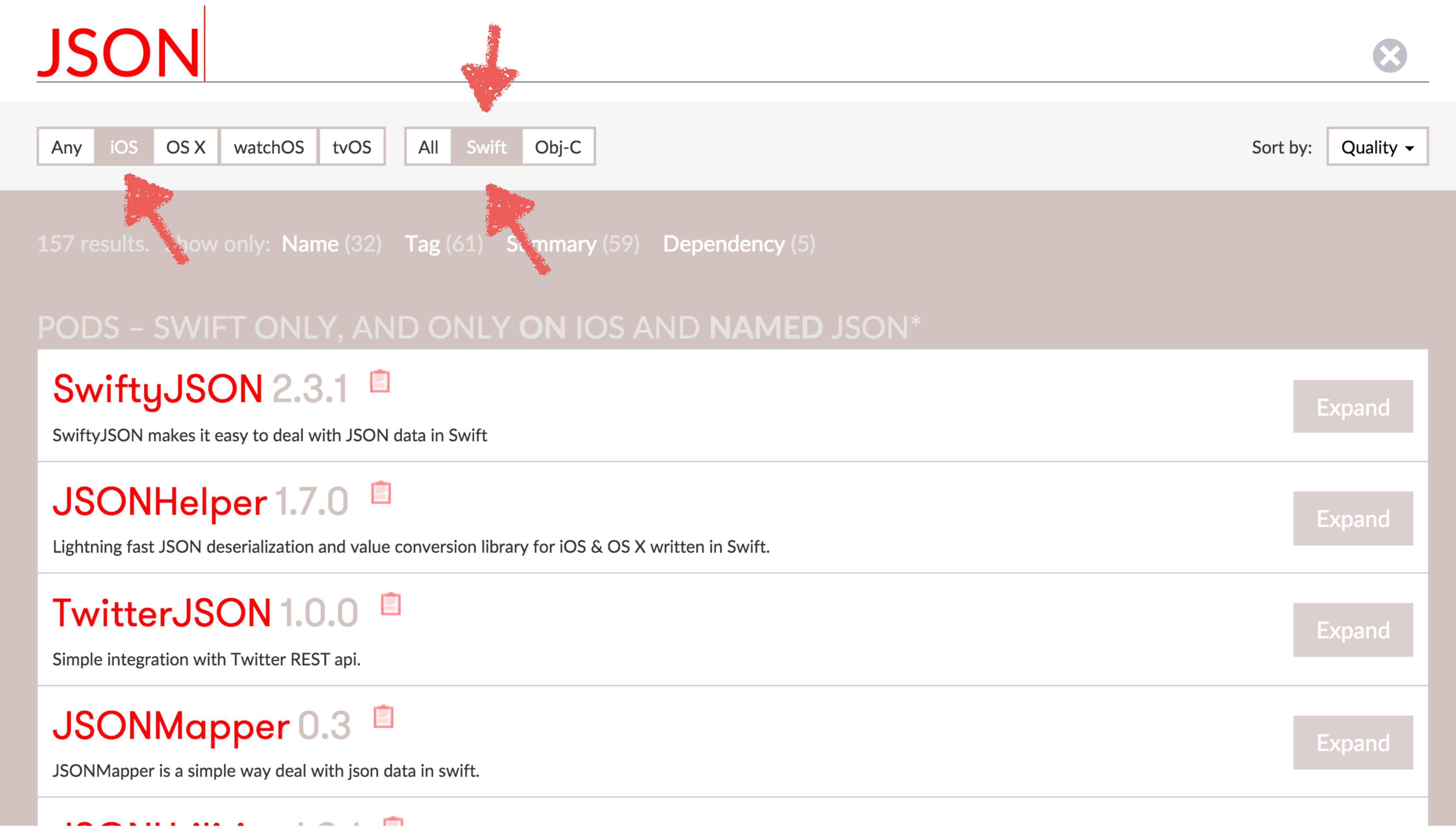
```
source ‘https://github.com/CocoaPods/Specs.git'

platform :ios, ‘8.0'
inhibit_all_warnings!

# Comment
pod 'SwiftyJSON', '2.1.3'

pod 'Alamofire', '~> 1.0'
```

CocoaPods

- `pod install`
  creates **workspace** File!
- A workspace can contain one or more XcodeProjects
- A Workspace is a container for XcodeProjects
- After using CocoaPods
  - you need to open the `xcworkspace`
    e.g. `SwiftParseExample`**`.xcworkspace`**
  - instead of the **xcodeproj**
    e.g. `SwiftParseExample`**`.xcodeproj`**
- `xcworkspace` is an indicator of CocoaPods usage



```
▼ SwiftParseExample                          M
  2 targets, iOS SDK 8.2
  ▼ SwiftParseExample
      AppDelegate.swift
      ViewController.swift                   M
      Main.storyboard
      Images.xcassets
      LaunchScreen.xib
    ▶ Supporting Files
  ▶ SwiftParseExampleTests
  ▶ Products
  ▶ Pods                                     ?
  ▶ Frameworks
▼ Pods                                       ?
  2 targets, iOS SDK 8.2
    Podfile                                  ?
  ▶ Frameworks
  ▶ Pods                                     ?
  ▶ Products
  ▶ Targets Support Files                    ?
```


JSON

Any | iOS | OS X | watchOS | tvOS

All | Swift | Obj-C

Sort by: Quality

157 results. Show only: Name (32) Tag (61) Summary (59) Dependency (5)

PODS – SWIFT ONLY, AND ONLY ON IOS AND NAMED JSON*

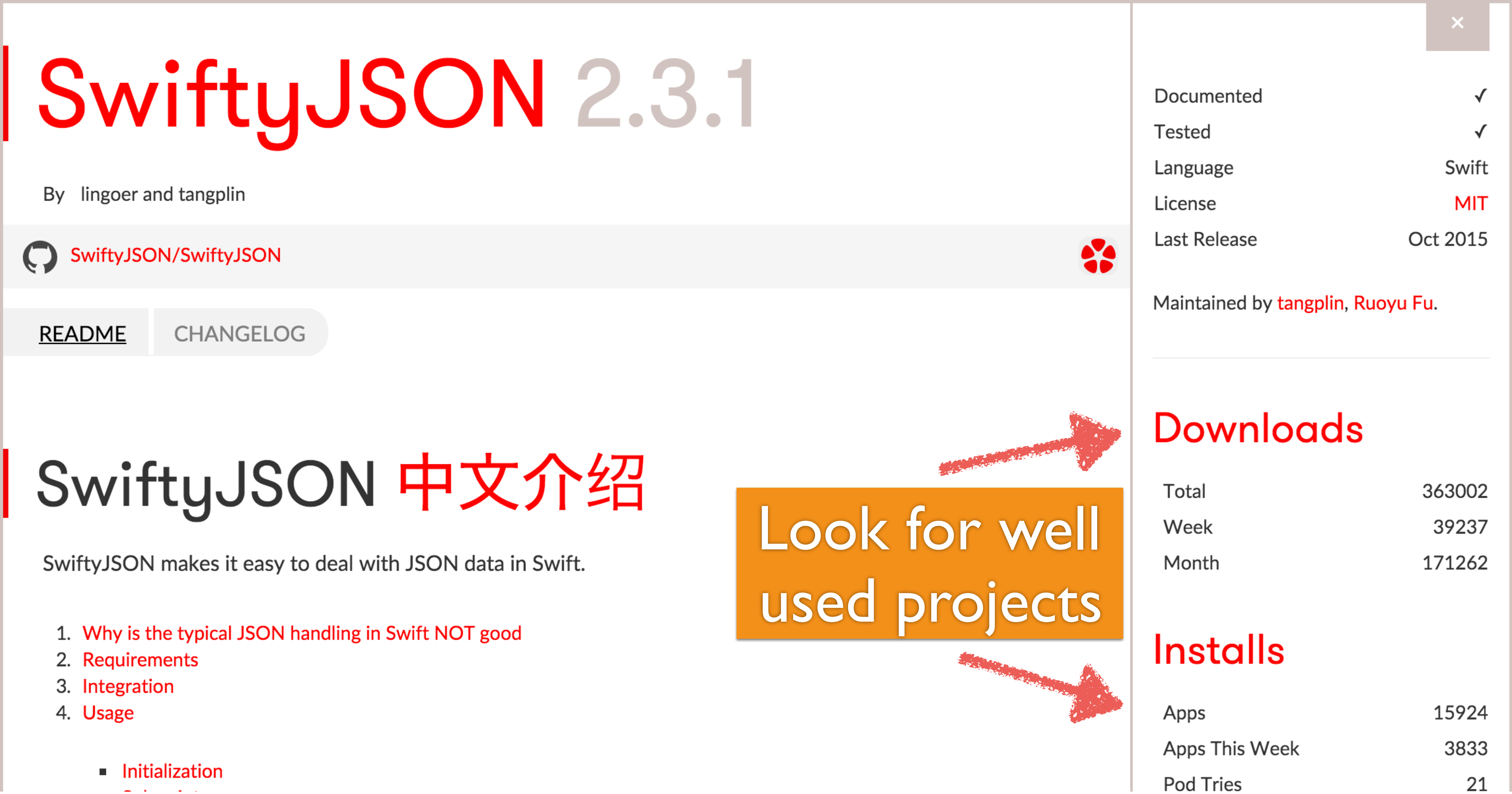
SwiftyJSON 2.3.1
SwiftyJSON makes it easy to deal with JSON data in Swift
Expand

JSONHelper 1.7.0
Lightning fast JSON deserialization and value conversion library for iOS & OS X written in Swift.
Expand

TwitterJSON 1.0.0
Simple integration with Twitter REST api.
Expand

JSONMapper 0.3
JSONMapper is a simple way deal with json data in swift.
Expand

PODS – SWIFT ONLY, AND ONLY ON IOS AND NAMED JSON*



SwiftyJSON 2.3.1

By lingoer and tangplin

SwiftyJSON/SwiftyJSON

README CHANGELOG

SwiftyJSON 中文介绍

SwiftyJSON makes it easy to deal with JSON data in Swift.

1. Why is the typical JSON handling in Swift NOT good
2. Requirements
3. Integration
4. Usage
   - Initialization

Documented	✓
Tested	✓
Language	Swift
License	MIT
Last Release	Oct 2015

Maintained by tangplin, Ruoyu Fu.

Downloads

Total	363002
Week	39237
Month	171262

Installs

Apps	15924
Apps This Week	3833
Pod Tries	21

Look for well used projects

Carthage

- A Cartfile file describes the used libraries
- Example Cartfile:

```
# Comment
github "SwiftyJSON/SwiftyJSON" >= 2.1.2


github ‘Alamofire/Alamofire', '~> 1.0’
```